**Proyecto EDA — Módulo *Data Analyst*:**

**Memoria de:**

*Análisis de Datos Exploratorio sobre carreras relacionadas a Data Science*

*Autora: Jocelyn Stefany De Paz Castellanos*

# Descripción General:

Este proyecto se conforma a partir de la interrogante “¿Cuánto es capaz de ganar un Data Scientist o alguien de una rama afín?”, por lo cual se utilizó un Dataset alojado en la página web de Kaggle. Los datos contenidos en este Dataset han sido extraídos de la página web [Glassdoor](https://www.glassdoor.com/index.htm).

Datasets:

1. [Data Analyst Jobs](https://www.kaggle.com/andrewmvd/data-analyst-jobs)
2. [Data Scientist Jobs](https://www.kaggle.com/andrewmvd/data-scientist-jobs)
3. [Data Engineer Jobs](https://www.kaggle.com/andrewmvd/data-engineer-jobs)

Repositorio original de los Datasets: [data\_jobs\_data](https://github.com/picklesueat/data_jobs_data)

Autor de los Datasets: [picklesueat](https://github.com/picklesueat)

El Dataset final tiene las siguientes características:

* Posee 11 columnas
* Los datos nulos se manejan como -1
* Posee 11,976 registros
* Se han tratado los nombres y descripciones de las ofertas de trabajo para eliminar palabras no relevantes y disminuir el tamaño al momento de exportar a CSV (ya que de este CSV se han formado las gráficas en Power BI)

# Librerías utilizadas:

Para poder manipular la información contenida en estos datasets, se utilizaron las siguientes librerías:

* Matplotlib: para la visualización
* Country\_list: para obtener una lista de todos los países, utilizada al momento de extraer información de la columna “Location”
* Numpy: para hacer pruebas previas con gráficos
* Pandas: esencial para el manejo de los datasets
* Nltk: para poder utilizar las stopwords

# Importación de Datasets y Clasificación

Para poder manipular de forma más sencilla la información, en lugar de definir una dirección física de un ordenador se ha colocado el enlace de los datos en crudo alojados en Github. Al hacer esto, de hecho, se obtienen 4,000 registros más que no están en los CSV de Kaggle de las 3 carreras.

Como se menciona en la documentación del Notebook, se categorizan de una vez los registros dependiendo del Dataset en el que se encuentren para no perder de vista de donde viene cada uno de ellos.

# Fusión de los Datasets

Manejar la información por separado iba a dar muchos problemas y más trabajo, por lo que se hace un conglomerado de toda la información (motivo por el cual debían categorizarse los registros). Ya que todas las columnas en todos los datasets son iguales, se utiliza un concat para que los datos simplemente se sobrepongan uno sobre otro sin problemas.

Esta fusión da como resultado un total de 16 columnas sin contar el index.

# Limpieza de columnas y filas

La variable principal de interés es “Salary Estimate”, por lo cual se elimina toda aquella información que no suponga una mayor utilidad, o porque tenga una gran cantidad de valores ausentes.

Entre las columnas que se eliminaron, se encuentran: 'Size', 'Founded', 'Easy Apply', 'Competitors', 'Headquarters', 'Industry' y 'Revenue'.

Entre los registros que se eliminaron, son aquellos en donde el salario contiene las palabras “Per Hour” o contiene como valor -1. Se eliminan, ya que supone un mayor esfuerzo convertir aprox 14 valores de 12,000 registros a un salario promedio anual.

La dificultad acá radicaba en como poder detectar ciertas palabras o cadenas de texto dentro de los valores del Dataframe, ya que no quería recorrer todos los registros con un for, por lo cual busqué como detectar esas palabras y me encontré con el método de “contains”.

# Limpieza de datos

Ciertamente, esta fue la parte que me supuso una mayor dificultad, especialmente para el cálculo del Salario Promedio Anual. Tardé aproximadamente 1 semana en buscar la forma de quitar todos aquellos caracteres que no me eran de utilidad para quedarme únicamente con los números.

Entre los problemas que tuve fue:

* Errores al momento de usar replace (trataba de usarlo como diccionario o como lista, pero no funcionaba, o trataba de poner muchos al mismo tiempo en un solo replace).
* Una vez los replace funcionaban, empecé a tener problemas con el Split ya que me decía que algunos valores no se podían “desempacar”. Es aquí donde me di cuenta de que los valores -1 me estaban suponiendo una dificultad, por lo que decidí eliminar todos aquellos registros que tuvieran este valor para que el Split funcionara correctamente con cada uno de los registros del Dataframe.
* Traté de realizar las modificaciones directamente en el Dataframe original, pero tenía advertencias de que no podía hacerlo, así que decidí por usar una copia de los valores de “Salary Estimate”
* Al tratar de obtener el promedio, como no me había percatado del problema de los valores con -1, siempre obtenía errores como que no era posible convertir el valor a un tipo float y no sabía porqué hasta que revisé de forma general los valores que se contenían en la columna y encontré registros con -1 que no habían sido tratados.

Luego de terminar de procesar los datos contenidos en esta columna, se realizó una pequeña limpieza a los nombres de las compañías donde simplemente se eliminaban números que estaban a la par del nombre de cada compañía (que, en teoría, es el rating de la compañía).

Finalmente, otra columna que supuso un gran desafío fue la de “Location”, ya que algunos campos tenían más de dos comas, y otros solamente tenían más de una. Y ya que casi todos los datos provenían de estados de USA, se evaluaba el estado y el país dependiendo de si aparecía en alguno de los listados de los estados de USA. De lo contrario, se evaluaba si aparecía en el listado de los países (se importa la librería para no agregar uno por uno en una lista) y si era así, entonces se definía ese valor como país y el estado como “unknow” o desconocido.

Entre los principales problemas que encontré, fue que algunos valores no tenían ninguna coma, y otros tenía más de dos, por lo que hacer un Split normal daba muchos errores.

Para eliminar aquellos registros que no tenían ninguna coma, primero se evaluaba si estaban presentes en el listado de países o de estados de USA. Dependiendo de en donde se encontraban, se le colocaba una coma a la izquierda o a la derecha.

Finalmente, para determinar el país de cada registro, se evaluaba si solamente tenía una coma o dos. Las que tenían dos comas, tenían información más concreta (por ejemplo, nombre del pueblo, la ciudad y el estado) de la cual no toda era necesaria ya que interesaba mayormente el nombre del estado. Por ello, se tomaba en cuenta la cantidad de valores que el Split regresaba, ya que, si era mayor de 3, significaba que tenía estos datos específicos que no eran de utilidad por lo que se tomaba la información del final (que era mayormente el acrónimo del estado).

La última limpieza que se realizó fue en torno a los nombres de los trabajos y sus descripciones. Esto fue mayormente para disminuir el peso del CSV que se utilizaría para el dashboard en Power BI, ya que sin tratar estos datos pesaba casi 50mb, pero al eliminar las stopwords llegó a pesar 33mb.

# Graficando los datos

La mayor dificultad en esta parte radicó mayormente en qué tipos de gráficos se podrían utilizar para representar la información y cómo utilizarlos.

Ya que lo que se pretendía era realizar una comparación entre diferentes variables, se optó por utilizar gráficos de barra para poder observar de forma clara las diversas diferencias que existían entre los datos. Sin embargo, saber crear los gráficos agrupados puede llegar a ser un poco “engañoso” especialmente por la ubicación de cada barra y la ubicación del texto en estas.

Pese a que el mayor análisis está en Power BI, también se hicieron evaluaciones previas sobre los datos para saber más o menos como era su comportamiento y como se comportaba el salario en cada una de ellas en el notebook que sirvieron de orientación al mismo tiempo para poder crear más gráficos en Power BI que podían ilustrar esas variaciones y diferencias entre las variables con respecto al salario.

# Documentación utilizada:

* <https://matplotlib.org/>
* <https://stackoverflow.com/>
* <https://www.tutorialgateway.org/python-matplotlib-bar-chart/>
* <https://www.delftstack.com/howto/matplotlib/how-to-rotate-x-axis-tick-label-text-in-matplotlib/>
* <https://www.youtube.com/watch?v=ZjQCPMO7LBE>
* <https://stackabuse.com/rotate-axis-labels-in-matplotlib/#:~:text=Rotate%20X%2DAxis%20Labels%20in%20Matplotlib&text=There%20are%20two%20ways%20to,set_xticklabels()%20and%20ax>.
* [www.geeksforgeeks.org](http://www.geeksforgeeks.org)
* <https://moonbooks.org/Articles/How-to-change-the-size-of-axis-labels-in-matplotlib-/>
* <https://cumsum.wordpress.com/2020/08/30/matplotlib-valueerror-shape-mismatch-objects-cannot-be-broadcast-to-a-single-shape/>
* <https://www.analyticsvidhya.com/blog/2020/09/colormaps-matplotlib/>
* <https://pythonspot.com/matplotlib-scatterplot/>
* <https://www.geeksforgeeks.org/how-to-get-rows-index-names-in-pandas-dataframe/>
* <https://www.codeforests.com/2020/07/18/calculate-percentage-within-group/>
* <https://www.digitalocean.com/community/tutorials/data-analysis-and-visualization-with-pandas-and-jupyter-notebook-in-python-3>
* <https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.scatter.html>
* <https://github.com/pandas-dev/pandas/issues/17605>
* <https://github.com/pandas-dev/pandas/issues/4493>